THE HAGUE UNIVERSITY OF APPLIED SCIENCES

KB-74 Presentation

OPSCHALER

Megan Meezen
Daan Boesten
Brian de Keijzer
Victor Gómez
Victor García
Pol de Visser

OPSCHALER

13-01-2019

Forecasting residential gas consumption with machine learning algorithms on weather data

General introduction

Why?

- Insights in reducing gas consumption
- Grid balancing
- Little research has been done

OPSCHALER

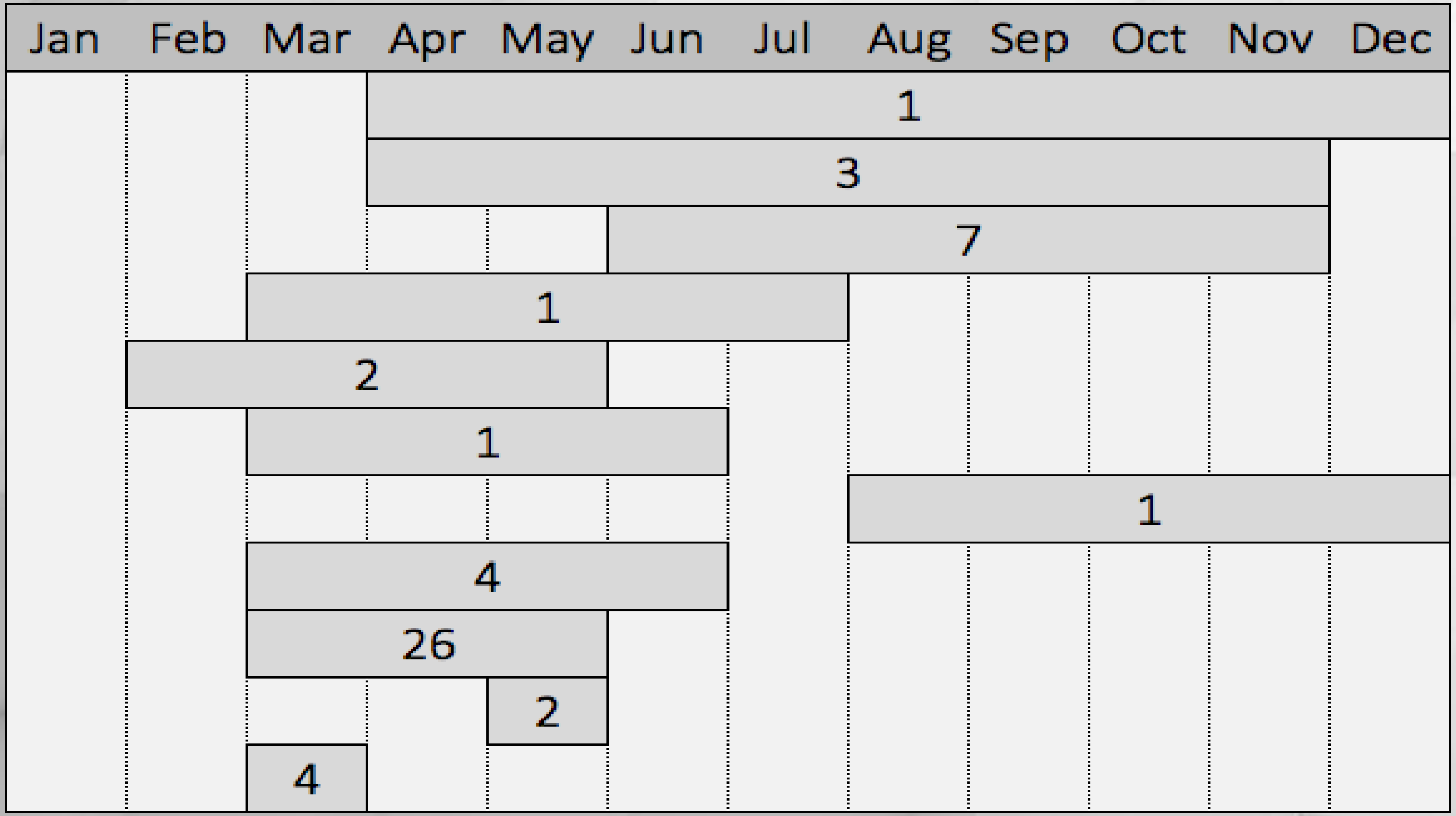
Methodology

How?

- Several machine learning algorithms have been compared
- Use as less features as possible
- Combining the following datasets:
  Nine months of smart meter data, provided by OPSCHALER (1 hour resolution)
  Weather data provided by KNMI (15 minutes resolution)

Available data

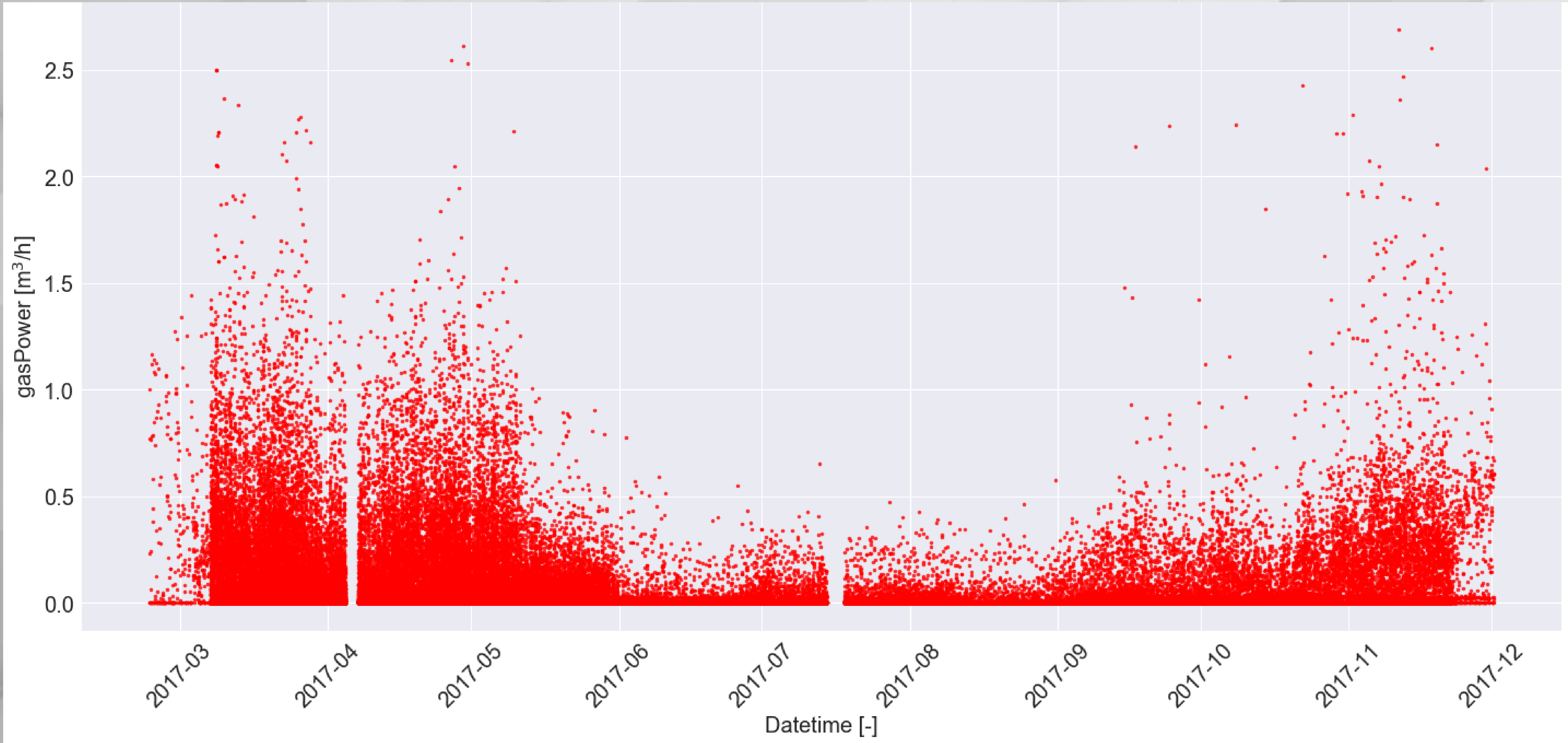
Distribution of the data acquisition from all dwellings.



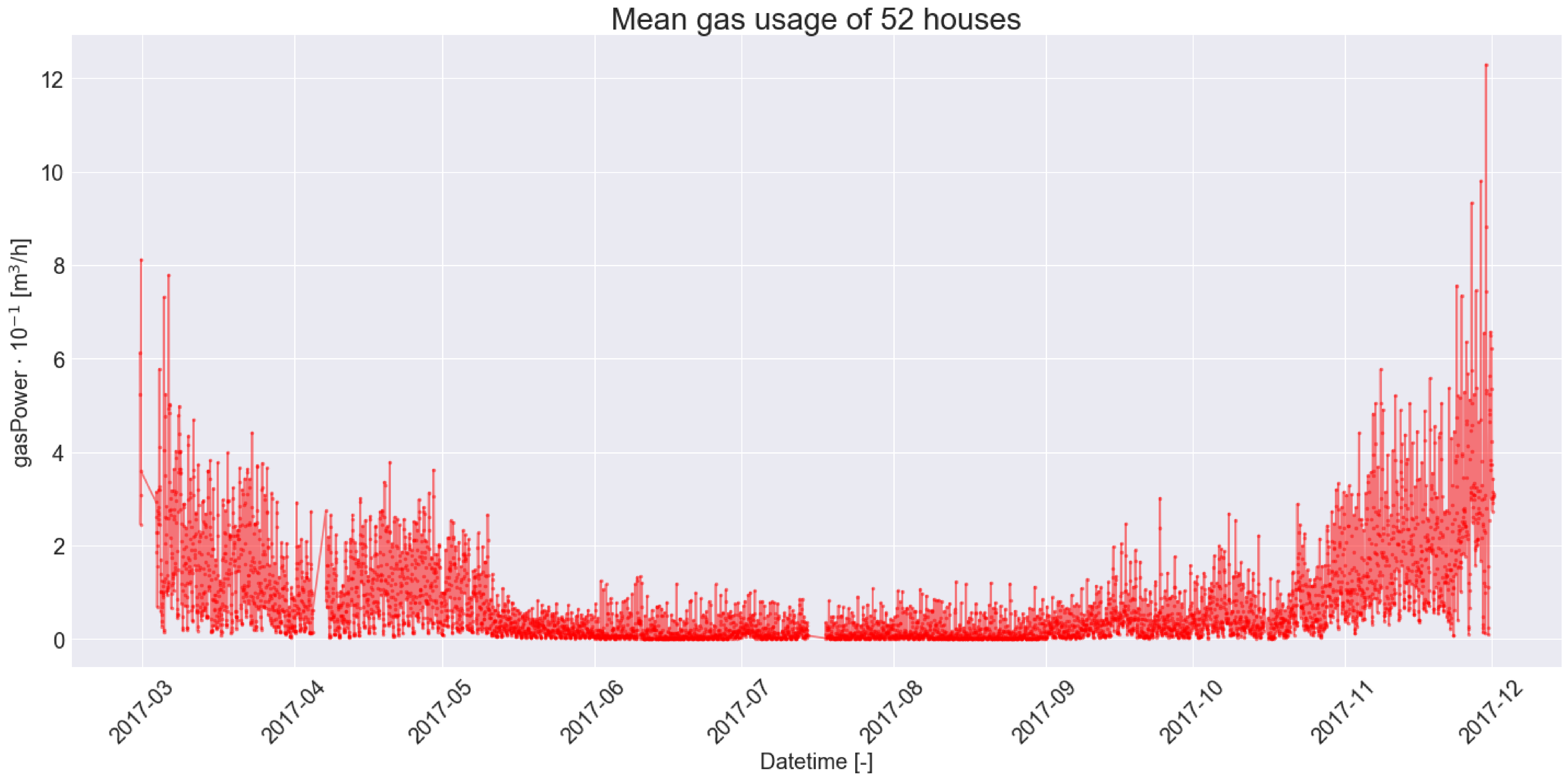
[4]

OPSCHALER

Individual house data



[5]


Mean gas usage of 52 houses
gasPower · 10^{-1} [m^3/h]
0
2
4
6
8
10
12
2017-03
2017-04
2017-05
2017-06
2017-07
2017-08
2017-09
2017-10
2017-11
2017-12
Datetime [-]

[6]

OPSCHALER

Data selection

- Parameter with highest Pearson correlation coefficient P with the target gasPower has been selected, which is the outside temperature T.
- Other parameters that suffix $|P| < 0.2$ with T are used as additional features.

OPSCHALER

Parameter	Unit	Description	Sample rate
DD	deg	Wind direction	15 min
DR	s	Precipitation time	
FX	m/s	Maximum gust of wind at 10 m	
FF	m/s	Windspeed at 10 m	
N	okta	Cloud coverage	
P	hPa	Outside pressure	
Q	W/m^2	Global radiation	
RG	mm/h	Rain intensity	
SQ	min	Sunshine duration	
T	°C	Temperature at 1,5 m (1 minute mean)	
T10	°C	Minimum temperature at 10 cm	
TD	°C	Dew point temperature	
U	-	Relative humidity at 1,5 m	
VV	m	Horizontal sight	
WW	-	Weather- and station-code	
Timestamp	-	Timestamp of data telegram (set by smart meter) in local time	10 s
eMeter	kWh	Meter reading electricity delivered to client, normal tariff	
eMeterReturn	kWh	Meter reading electricity delivered by client, normal tariff.	
eMeterLow	kWh	Meter reading electricity delivered to client, low tariff	
eMeterLowReturn	kWh	Meter reading electricity delivered by client, low tariff	
ePower	kWh	Actual electricity power delivered to client	
ePowerReturn	kWh	Actual electricity power delivered by client	
gasTimestamp	-	Timestamp of the gasMeter reading (set by smart meter) in local time	1 h
gasMeter	m^3	Last hourly value (temperature converted), gas delivered to client	
gasPower	m^3/h	Difference between current and previous gasMeter value *	
hour of day	-	Hour of day from the timestamp *	
day of week	-	Day of week when sample has been acquired *	
season	-	Season of the year when sample has been acquired *	

[8]

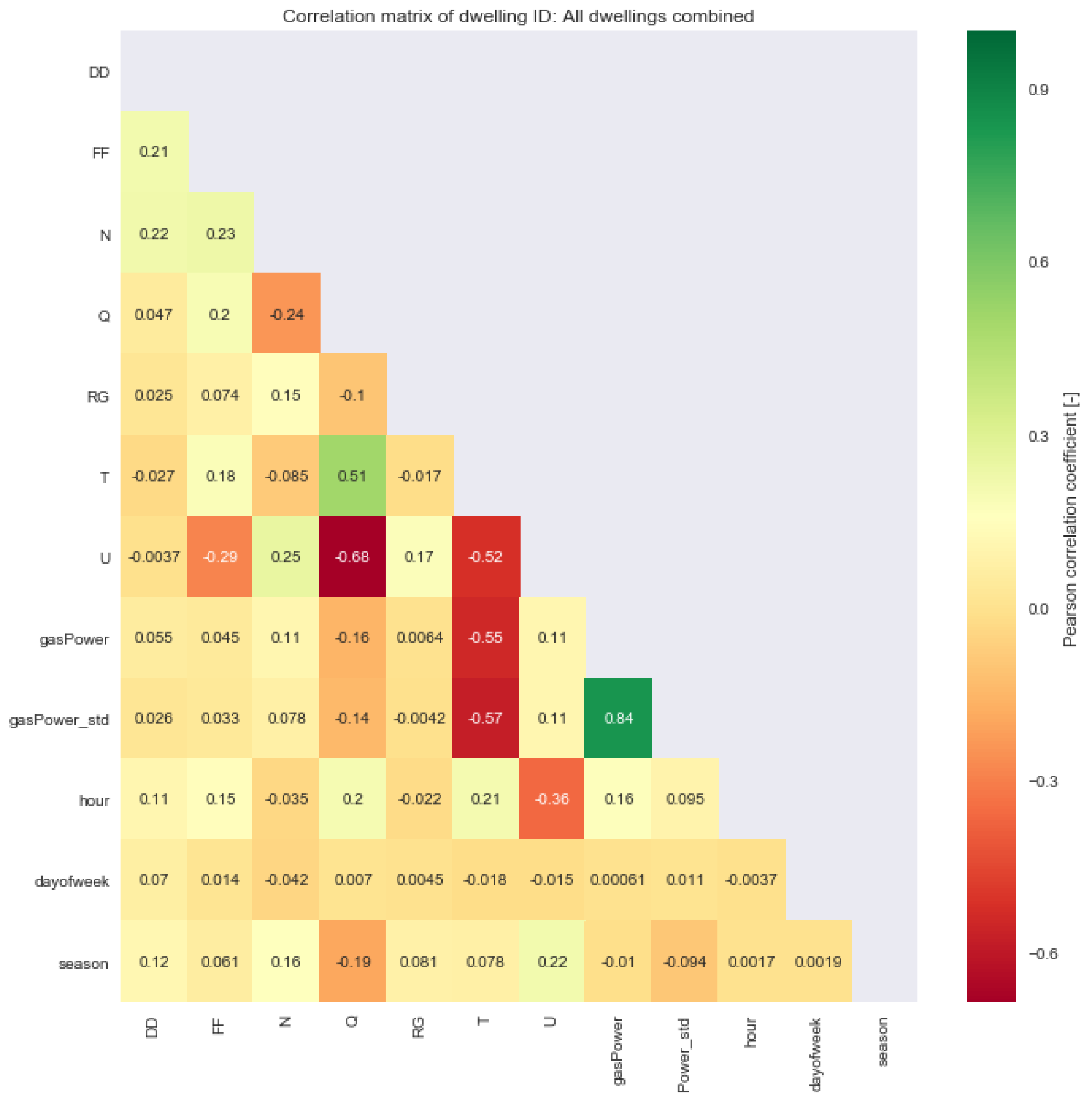


Figure 14. The Pearson correlations of all the available parameters.

OPSCHALER

Used data

Selected features:

FF	Windspeed (m/s)
RG	Rain intensity (mm/h)
T	Outside temperature (°C)
Hour of day	Current hour of the day
Day of week	Current day of the week
Season	Current season

To predict the target:

Gaspower	Gas consumption of that hour $[\mathrm{m}^3/\mathrm{h}]$

Note that everything is on a one hour resolution.

[10]

OPSCHALER

Train, validation & test dataset

- **Train size** - 50 %
- **Validation size** - 20%
- **Test size** - 30 %

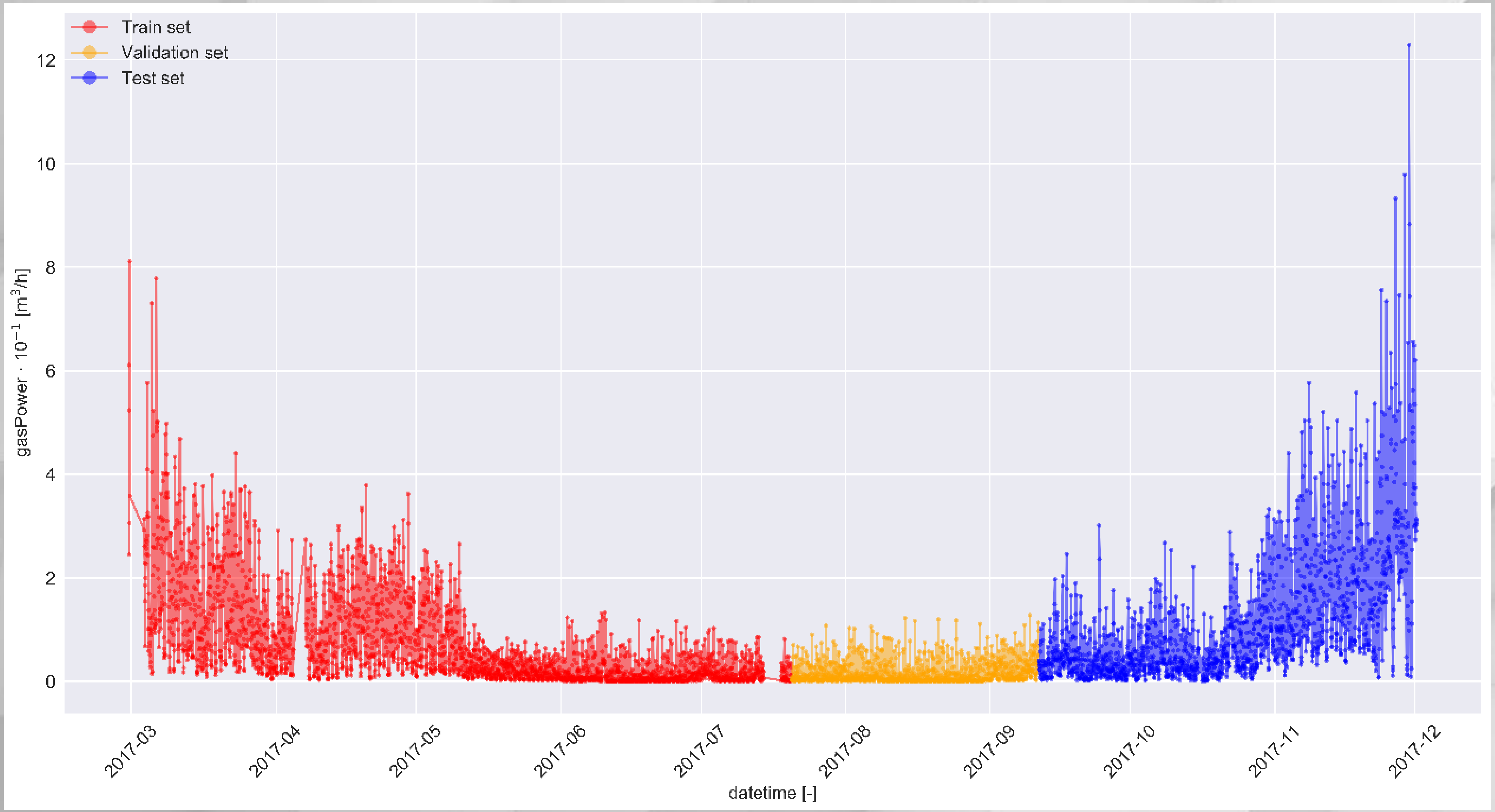


Figure 2. A visualization of the train, validation and test dataset distribution on a daily resolution.

OPSCHALER

Model evaluation

Loss function:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

Other evaluation metrics used are :

$$\mathrm{MAPE} = \frac{100\%}{n}\sum_{i=1}^{n}\frac{|\hat{Y}_i - Y_i|}{|Y_i|}$$

$$\mathrm{SMAPE} = \frac{100\%}{2n}\sum_{i=1}^{n}\frac{|Y_i - \hat{Y}_i|}{|\hat{Y}_i| + |\hat{Y}_i|}$$

[13]

OPSCHALER

Optimizers

Adam & Nadam have been used with a cosine annealing learning rate, defined as:

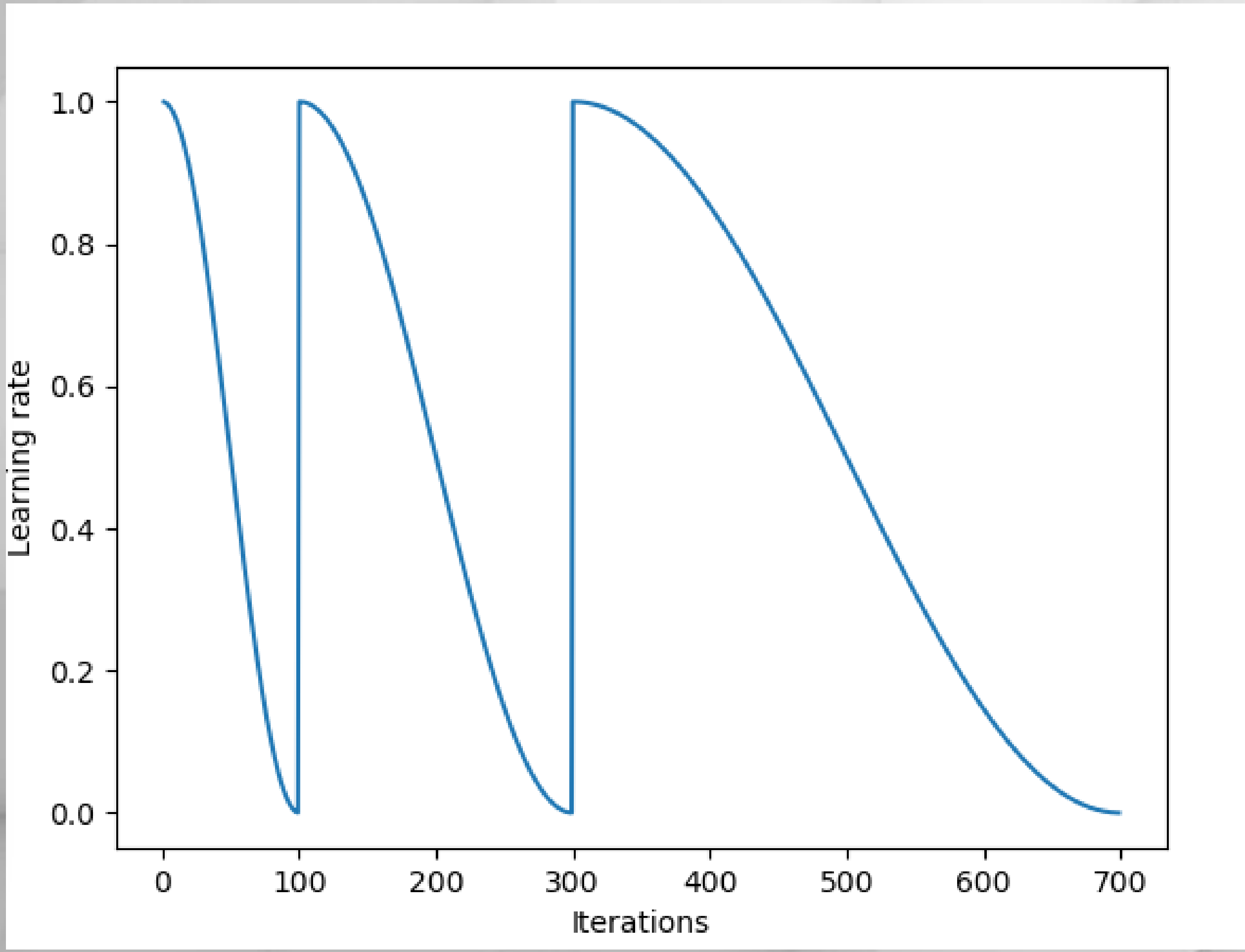
$$\eta_t = \eta^i_{min} + \frac{1}{2}(\eta^i_{max} - \eta^i_{min})\left(1 + \cos\left(\frac{T_{cur}}{T_i}\pi\right)\right)$$

Where η^i_{max} and η^i_{min} are the ranges for the learning rate and T_{cur} is the number of epochs since the last restart, with T_i the total amount of epochs.

OPSCHALER

Optimizers

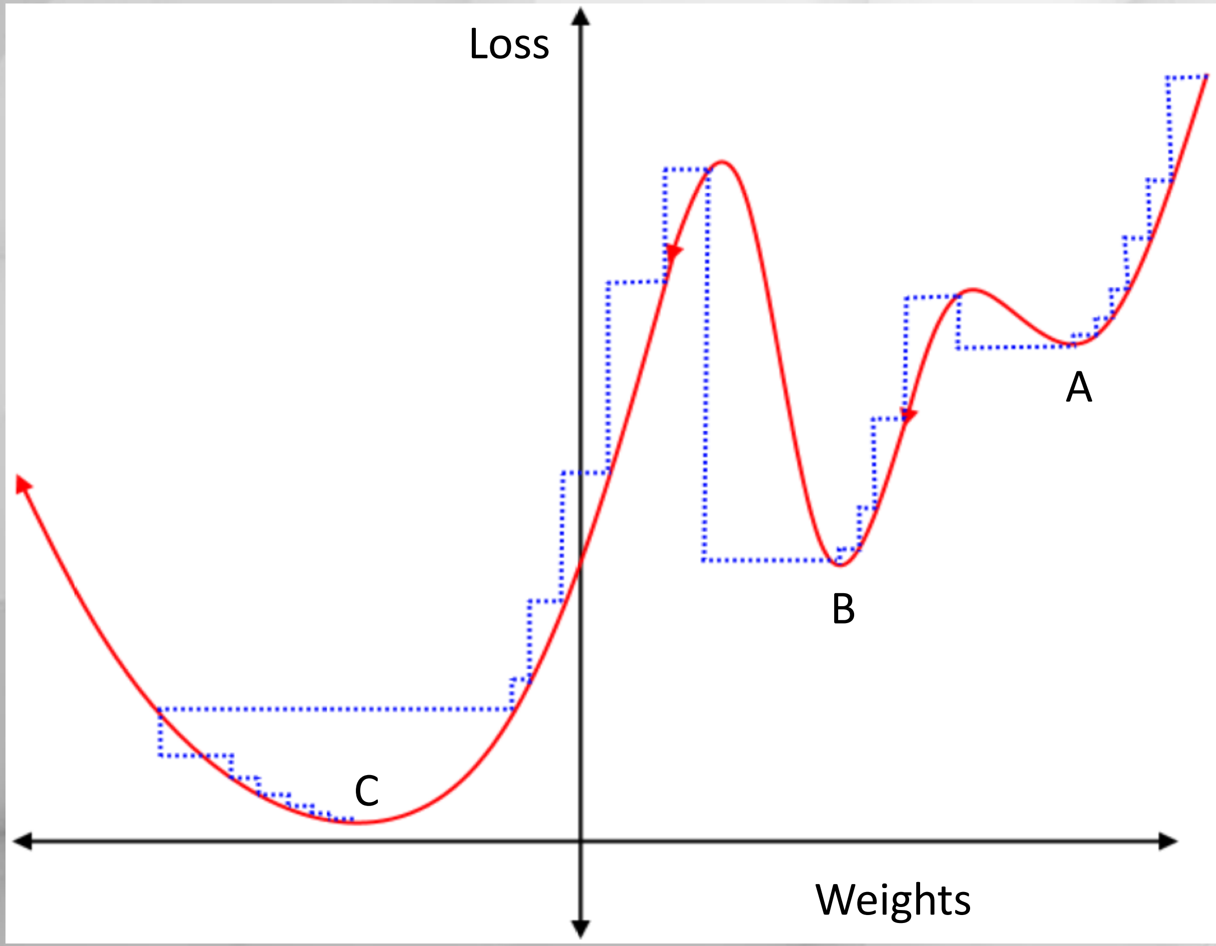
Thee epochs visualized:



[15]

Optimizers

The effect of a cosine annealing learning rate visualized in 2D



OPSCHALER

(Batch) normalization & one hot encoding

The features & weights after each layer, apart from the output layer, are normalized.

Hour of the day, day of the week and season are one hot encoded.

$$\begin{bmatrix} 0 \\ 1 \\ 2 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Representing the features extracted from the timestamp this way allows the models to assign different weights to for example 07:00 AM on a Monday and 09:00 AM on a Saturday.

OPSCHALER

Architecture evaluations

(Hyper) parameters & (random) architecture space searches

```
for _ in range( {{ choice([0, 1, 2, 4, 8, 16]) }} ):
        Dense( {{ choice([8, 16, 32, 64, 128, 256, 512, 1024]) }}
        ....
        Dropout( {{ uniform([0, 1]) }} )
        ...

# repeat for n layers
```

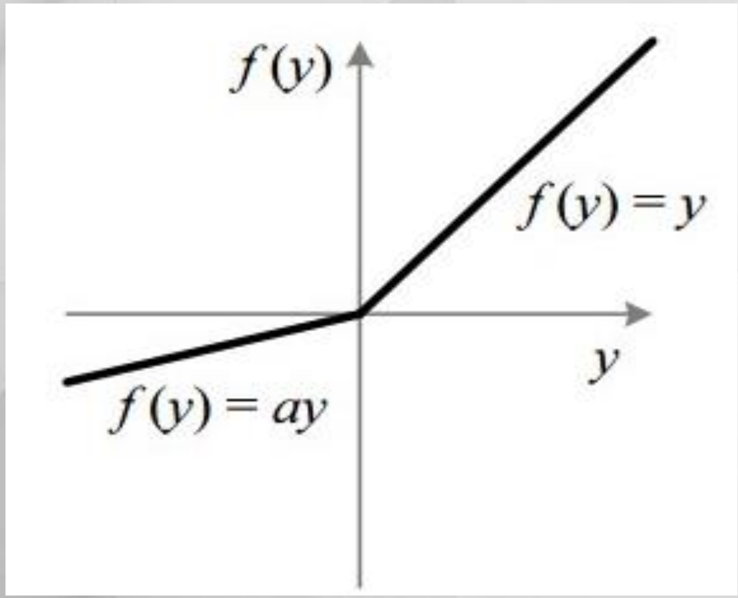

OPSCHALER

Initial neural network setup

GPU: NVIDIA GeForce 960m

To minimize vanishing & exploding gradient:

Weight initialization:	Truncated normal distribution
Batch normalization:	After each layer (apart from the output)
Activation function:	LeakyRelu



OPSCHALER

The used models

Multivariate Linear Regression

$$y = b_0 + b_1 X_0 + b_2 X_1 + \ b_3 X_2 + \sum_{i=0}^{23} b_{4+i} X_{3+i} + \sum_{j=0}^{6} b_{28+j} X_{27+j} + \sum_{k=0}^{3} b_{34+k} X_{33+k}$$

Where X_0 ... X_{33+k} are all the used features

Deep neural network (DNN)

Input data

$$X_{train} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

$[1 \quad 2 \quad 3]$

$[4 \quad 5 \quad 6]$

$[7 \quad 8 \quad 9]$

$$y_{train} = \begin{bmatrix} 10 \\ 20 \\ 30 \end{bmatrix}$$

Epoch finished: repeat the process

Adjusting weights

Validation

Deep Neural Network

$y_{predicted} = [25]$

$y_{predicted} = [32]$

$y_{predicted} = [32]$

This is a **simplified** representation of the training process.

[22]

OPSCHALER

Deep neural network (DNN)

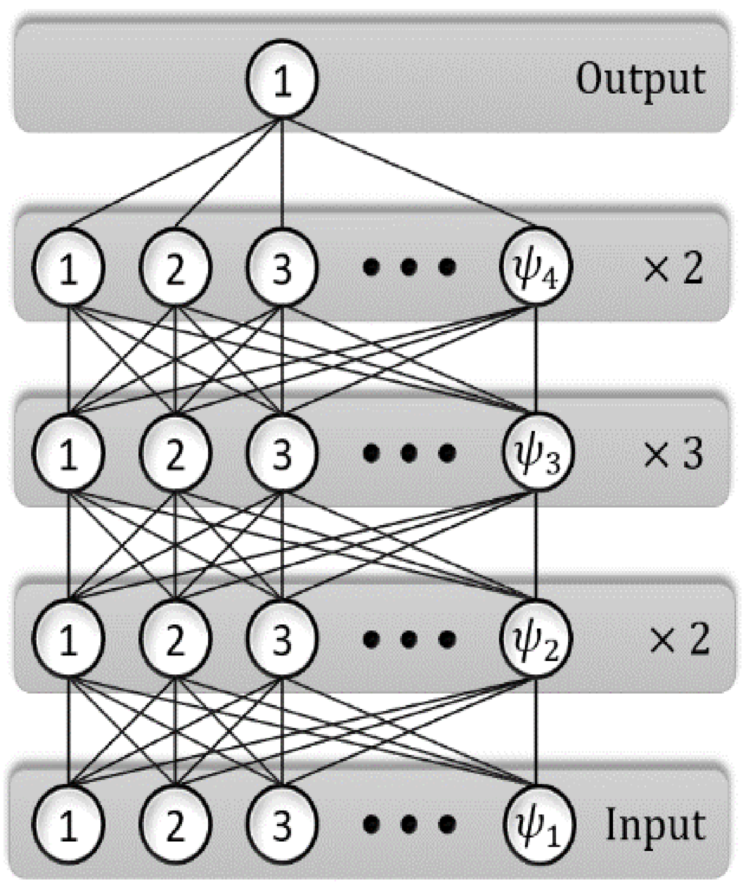


Figure 3. Where $\psi_1, \psi_2, \psi_3, \psi_4$ represent the number of nodes of respective layer and are equal to 64, 256, 64, 1024, 8 respectively. The $\times 2$ represents this layer configuration being repeated two times behind each other.

[23]

OPSCHALER

Sequential: Recurrent Neural Network (RNN)

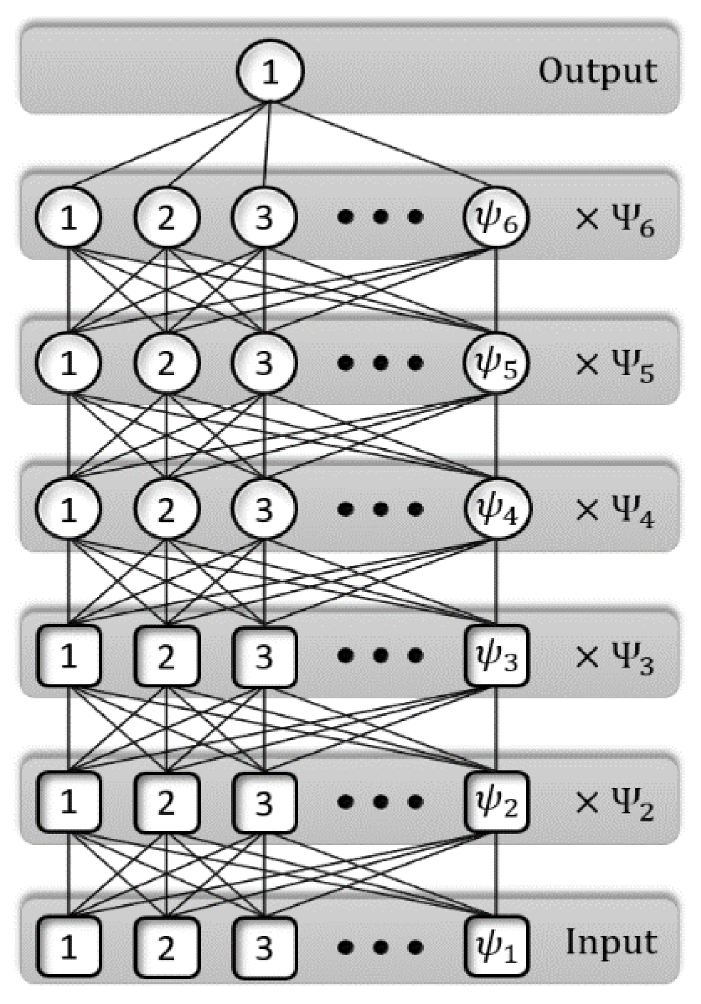


Figure 4. Where $\psi_1, \ldots, \psi_6$ represent the number of nodes of respective layer. For LSTM these are equal to 8, 0, 16, 128, 8, 16 and for GRU are equal to 16, 8, 4, 0, 8, 8 respectively.
Each layer configuration being repeated Ψ_i times behind each other is represented by $\times \Psi_i$, where i is the layer number. For LSTM $\Psi_2, \ldots, \Psi_6$ are equal to 0, 1, 3, 2, 1 and for GRU are equal to 1, 1, 0, 4, 1 respectively.

[24]

Sequential RNN: Input data

- Looking back three timesteps.
- Predicting one step ahead (this could made to predict more timesteps ahead)

Input data

$$X_{train} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \\ 13 & 14 & 15 \\ 16 & 17 & 18 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \begin{bmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix} \begin{bmatrix} 7 & 8 & 9 \\ 10 & 11 & 12 \\ 13 & 14 & 15 \end{bmatrix}$$

$$y_{train} = \begin{bmatrix} 10 \\ 20 \\ 30 \\ 40 \\ 50 \\ 60 \end{bmatrix}$$

Validation

Epoch finished: repeat the process

Adjusting weights

RNN

$$y_{predicted} = [43]$$

$$y_{predicted} = [55]$$

$$y_{predicted} = [67]$$

Notice how $y_{train} = [10, 20, 30]$ cannot be predicted.
This is a **simplified** representation of the training process.

OPSCHALER

A closer look at the data

$$X_{train} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \\ 13 & 14 & 15 \\ 16 & 17 & 18 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad \begin{bmatrix} 4 & 5 & 6 \\ 7 & 8 & 9 \\ 10 & 11 & 12 \end{bmatrix} \quad \begin{bmatrix} 7 & 8 & 9 \\ 10 & 11 & 12 \\ 13 & 14 & 15 \end{bmatrix}$$

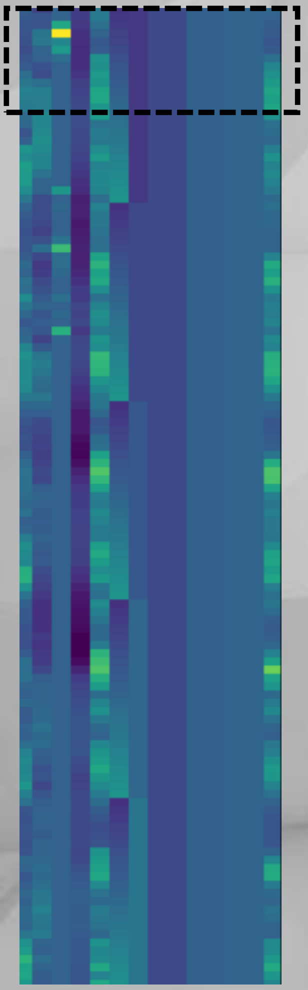
$$\begin{bmatrix} 10 \\ 20 \\ 30 \\ 40 \\ 50 \\ 60 \end{bmatrix} = y_{train}$$

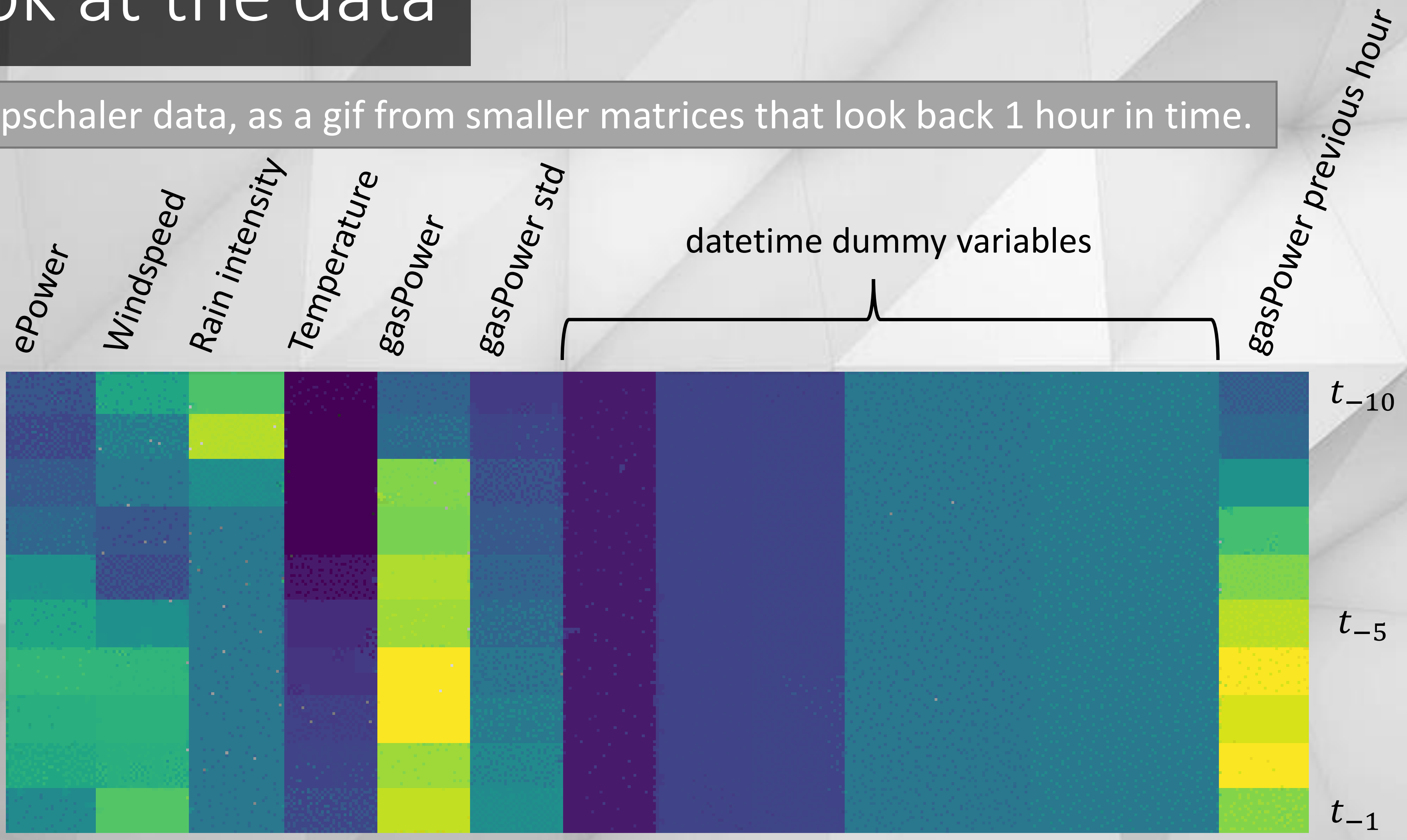
What if we interpret these 3x3 matrices as ‘images’?

OPSCHALER

A closer look at the data

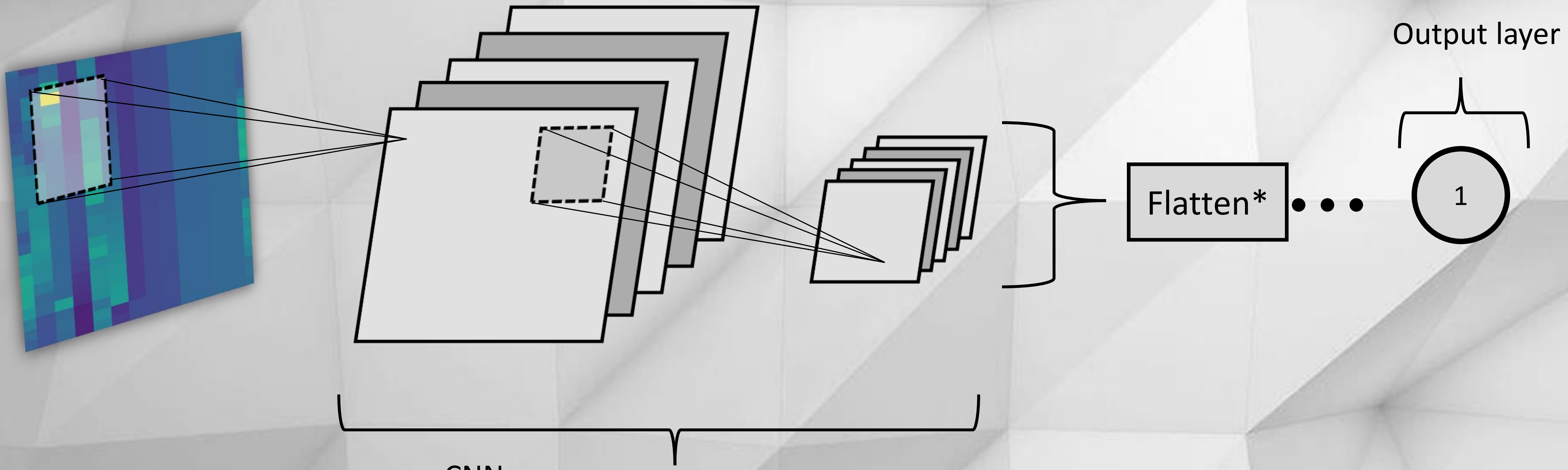
120 hours of Opschaler data, as a gif from smaller matrices that look back 1 hour in time.





OPSCHALER

Sequential: Convolutional Neural Network (CNN)



CNN:
- Multiple filter
- Different filter sizes
- Pooling
- Lots of combinations possible

*Reshapes an array from for example (4,4) to (16,)

Sequential: Convolutional Neural Network (CNN)

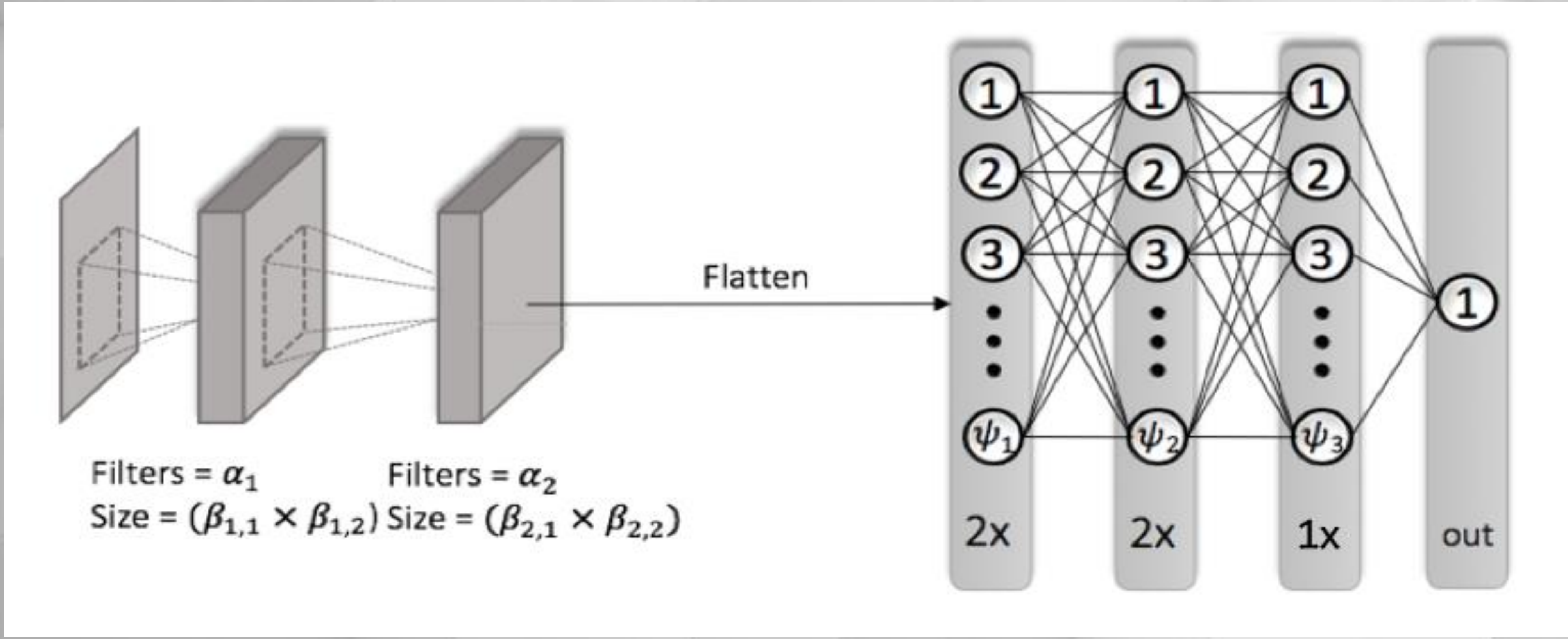
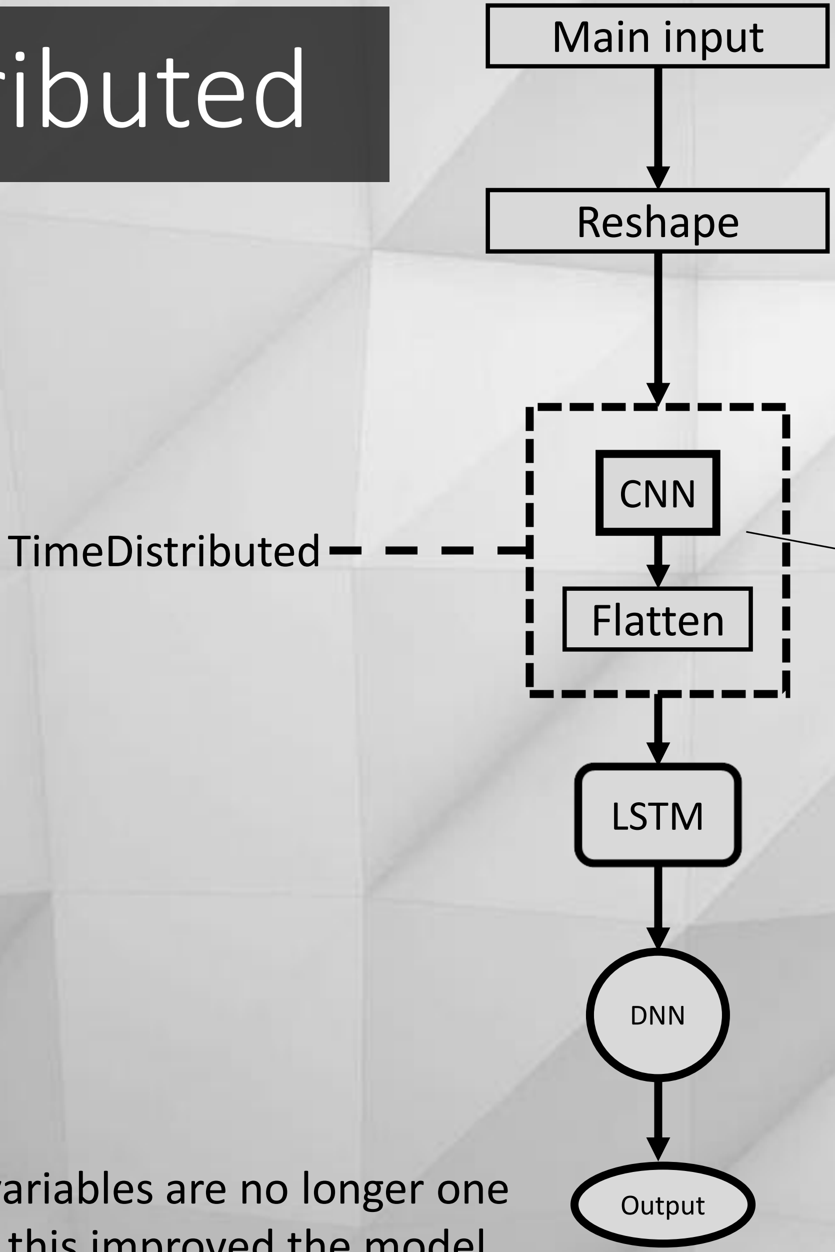


Figure 5. Where α_1 is equal to 5, α_2 is equal to 8, $(\beta_{1,1} \times \beta_{1,2})$ equals (8 x 4) and $(\beta_{2,1} \times \beta_{2,2})$ equals (10 x 8). The final output of the CNN is flatted and fed into a DNN where $\psi_1 \ldots \psi_3$ equals 64, 128, 256 and $\Psi_1 \ldots \Psi_3$ are equal to 2, 2, 1 respectively.

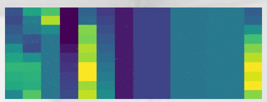
OPSCHALER

Time distributed



(120, 9) → (height, width) or (timesteps, features)

(5, 24, 9, 1) → (N images, height, width, channels*)

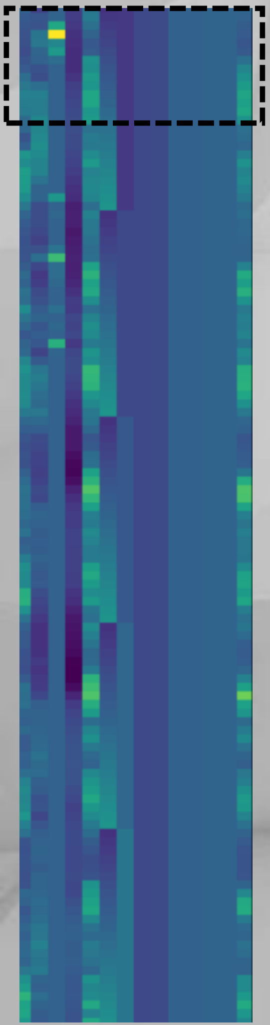


* 3 for RGB, 1 for greyscale images

CNN input: 5x an image of (24,9,1)

TimeDistributed output: 5x flattened CNN output

The idea is to make the RNN see a sequence of the CNN ‘scanning’ the original (120,9) image five times.



The dummy variables are no longer one hot encoded, this improved the model performance.

[30]

OPSCHALER

Results

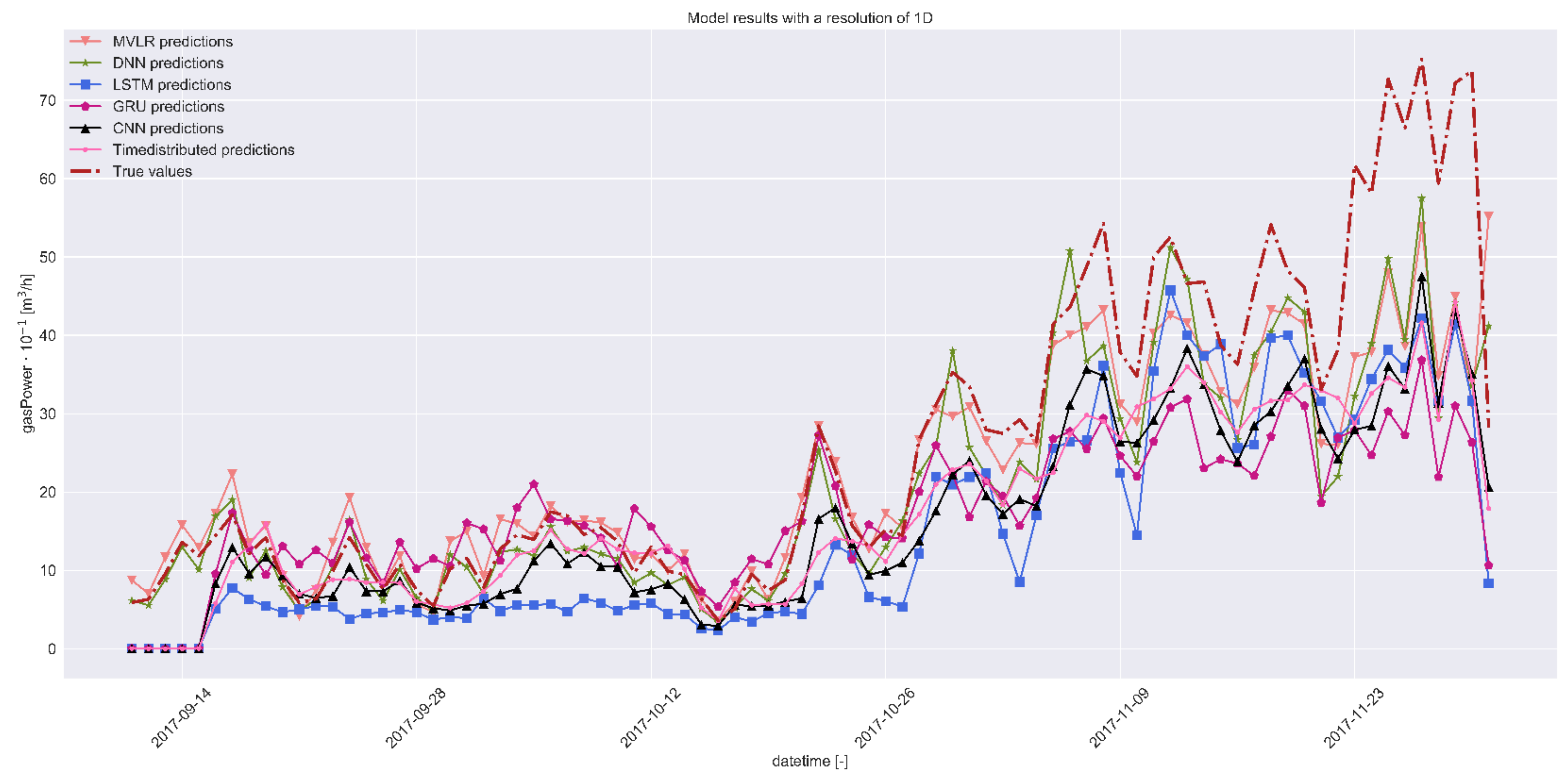


Figure 6. The forecasted gasPower consumption of the different models on a daily resolution.

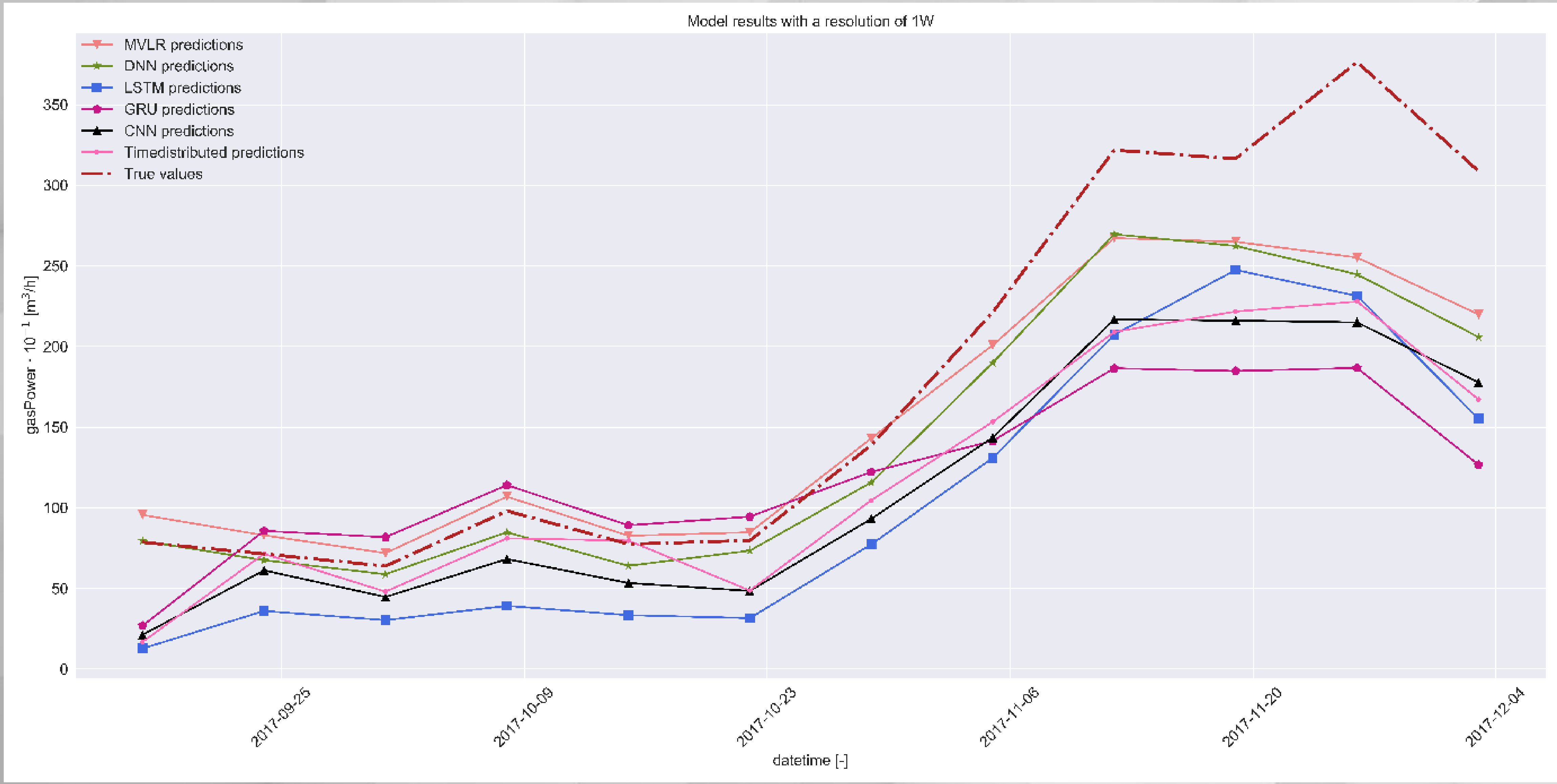


Figure 7. The forecasted gasPower consumption of the different models on a weekly resolution.

OPSCHALER

Model [-]	Resolution	MSE [-]	MAPE [%]	SMAPE [%]	Architecture evaluations [-]	time per epoch [s]	Epochs [-]
MVLR	Hour	**0.62**	78.3	19.3	n.a.	n.a.	n.a.
	Day	**99.0**	**20.2**	**9.20**			
	Week	$\mathbf{2.44 \cdot 10^3}$	**17.0**	**7.80**			
DNN	Hour	0.67	**50.1**	**16.6**	$1.00 \cdot 10^3$	$4.00 \cdot 10^{-6}$	$3.50 \cdot 10^4$
	Day	104	25.1	10.5			
	Week	$2.96 \cdot 10^3$	20.1	8.70			
LSTM	Hour	1.00	139	33.9	50.0	$4.62 \cdot 10^{-3}$	$4.00 \cdot 10^3$
	Day	206	99.7	30.1			
	Week	$7.06 \cdot 10^3$	95.0	31.1			
GRU	Hour	1.19	78.6	30.5	100	0.11	$4.00 \cdot 10^3$
	Day	264	59.8	19.4			
	Week	$9.38 \cdot 10^3$	45.3	16.9			
CNN	Hour	0.84	84.3	28.3	50.0	0.76	$8.00 \cdot 10^3$
	Day	115	33.3	13.5			
	Week	$3.51 \cdot 10^3$	32.3	13.6			
Time Dist.	Hour	0.91	74.0	26.8	100	$2.88 \cdot 10^{-3}$	$4.00 \cdot 10^3$
	Day	184	42.7	16.4			
	Week	$5.93 \cdot 10^3$	41.5	16.3			

Conclusion

- DNN performs best on hourly predictions
- MVLR performs best on daily and weekly predictions

Recommendations

- Full year (or more) data should improve accuracy
- Predict on individual homes
- Use electricity consumption as feature

CLIMA 2019

Built environment facing climate change

REHVA 13th HVAC World Congress

26 - 29 May, Bucharest, Romania

OPSCHALER

Q & A

OPSCHALER